CLAIMS

What is claimed is:

1. A binary search tree method, comprising:

providing a set of items, each item having a respective value;

identifying a unique value in the set of values;

setting the unique value as a node in a binary search tree, the node having a first branch for receiving one of the values that is less than the unique value, and having a second branch for receiving one of the values that is greater than the unique value;

identifying a duplicate value in the set of values;

setting the duplicate value as an element in the binary search tree; and

extending a list branch from the element, the list branch identifying at least one of the items having a value equal to the duplicate value.

2. The method according to claim 1, further including the step of extending a subtree from the element, the node tree having a base node value equal to the duplicate value.

3. The method according to claim 1 wherein the element is a hook.

4. The method according to claim 1 further including associating address information with the element.

5. The method according to claim 4 further including:

negating the address information; and

storing the negated address information.

6. The method according to claim 1 wherein the element is identified by setting a flag bit in a data value associated with the element.

7. The method according to claim 1 wherein the element is identified by setting the element as a negative number having an absolute value equal to the duplicative value.

8. A method for managing a memory, comprising:

providing a set of available memory blocks, each memory block having a size;

requesting an allocation of memory;

searching for one of the available memory block that minimally satisfies the allocation request, the searching process further comprising:

comparing the size of the allocation request to an element in a binary search tree;

determining that the allocation request is minimally satisfied by the value of the element;

using a list extending from the element, the list identifying duplicative values equal to the value of the element; and

decrementing by one the number of duplicative value memory blocks listed at the element; and

allocating one of the memory blocks at the size equal to the value of the element.

9. The method according to claim 8, further comprising:

providing address data for each memory block;

retrieving a memory address associated with the element; and

determining that the memory address is a negative number.

10. The method according to claim 8, wherein the element is a hook in a binary search tree.

11. The method according to claim 8, wherein the element is identified as a negative number and the value of the element represents an absolute value.

12. The method according to claim 8, further including the step of arranging the sizes of the memory blocks into a binary search tree, the binary search tree further comprising nodes and hooks.

13. The method according to claim 12, wherein the hook has a branch for listing the duplicative values.

14. The method according to claim 12, wherein the hook has a branch for receiving a subtree.

15. A process for updating a binary search tree, comprising:

organizing a set of values in a binary search tree, each unique value being represented by a node element;

receiving a new value to be added to the set of values;

determining that the new value is duplicative of the value of one of the nodes;

identifying a subtree for the node having the duplicative value;

replacing the node having the duplicative value with a hook indicating the duplicative value;

extending the subtree from the hook, the subtree having a base parent node at the duplicative value; and

extending a list from the hook, the list identifying one of the duplicative values.

16. The process according to claim 15, wherein replacing the node with the hook includes changing the sign of a data point associated with the new value.

17. The process according to claim 16 wherein the data point is a memory address.

18. The process according to claim 15, wherein replacing the node with the hook includes changing the sign of the duplicative value.

19. The process according to claim 15, further including:

receiving a second new value to be added to the set of values;

determining that the second new value is duplicative of the value of the hook; and

adding an indication to the list that another duplicative value has been added to the set of values.

20. A handheld portable device, comprising:

RAM memory;

input and output subsystems;

an embedded processor; and

a binary search tree engine implement a memory management process further comprising:

receiving a request for an allocation of memory;

searching an available memory block that minimally satisfies the allocation request, the searching process further comprising:

comparing the size of the allocation request to an element in a binary search tree;

determining that the allocation request is minimally satisfied by the value of the element;

using a list extending from the element, the list identifying duplicative values equal to the value of the element; and

decrementing by one the number of duplicative value memory blocks listed at the element; and

allocating one of the memory blocks at the size equal to the value of the element.

21: The device according to claim 20, further comprising a transceiver and an antenna.

22. A method of searching in a binary search tree, comprising:

moving to a new element in a binary search tree, the element having a value;

retrieving a data flag associated with the new element;

determining whether the data flag is set;

identifying, responsive to the determining step, that the new element is a hook element, the hook element being capable of having two branches; and

making a value comparison to the value.

23. The method according to claim 22, further comprising:

providing a list of duplicate members on one branch of the hook; and

allocating, responsive to making the comparison, one member from the list of duplicate members.

24. The method according to claim 22, further comprising:

providing a list of duplicate members on one branch of the hook;

providing a subtree on the other branch of the hook; and

moving, responsive to making the comparison, to an element on the subtree.

25. The method according to claim 22, wherein the data flag is a memory address.

26. The method according to claim 22, wherein the data flag is the sign bit for a memory address.

27. The method according to claim 22, wherein the data flag is the sign bit for the value.

28. A binary search tree, comprising:

a node element having a node value and two branches, comprising:

a first branch having only values that are greater than the node value; and

a second branch having only values that are less than the node value; and

a hook element having a hook value and two branches, comprising:

a first branch having only values that are equal to the hook value; and

a second branch having values that are not equal to the hook value.

29. The binary search tree according to claim 28, wherein the second branch of the hook element includes a subtree.

30. The binary search tree according to claim 29, wherein the subtree has a root value equal to the hook value.